

Universitatea POLITEHINICA București
Facultatea de Automatică și Calculatoare

Managementul Proiectelor Software

Hierarchy Analysis
Software Development Document
v 1.0

Echipa Farik

București, 2012

0. Cuprins

0. Cuprins	2
1. Scopul documentului	3
2. Vedere de ansablu a solutiei proiectate	3
2.1. Fond	3
2.2. Context si design de baza	3
2.3. Decizii de implementare	3
2.4 Formatul fisierelor utilizate	3
3. Modelul datelor	4
3.1. Clasa HierarchyAnalysis	4
3.1.1. Structura clasei	4
3.1.2. Diagrama UI a clasei	4
3.2. Clasa Editor	5
3.2.1. Structura clasei.....	5
3.2.2. Diagrama UI a clasei	6
3.2.3. Diagrama de secventa a clasei	6
3.2.3. Diagrama UML a clasei	7
4. Structuri de date folosite	8
5. Planul de testare	8
5.1. Planul de testare	8
5.2. Managementul bug-urilor	8
5.3. Workflow-ul de fixare a bug-urilor	8
6. Cerinte de sistem	8

1. Scopul documentului

Acest document are rolul de a descrie soluția tehnică proiectată pentru componenta *Hierarchy Analysis* din cadrul sistemului OCR modular pe care studenții îl dezvoltă în cadrul cursului de Managementul Proiectelor Software din cadrul Facultății de Automatică și Calculatoare.

Documentul servește drept ghid pentru implementarea soluției software pentru modulul *Hierarchy Analysis*.

2. Vedere de ansamblu a soluției proiectate

2.1. Fond

Funcțiile și scopul acestui proiect sunt redactate în Documentul Specificațiilor de Proiect la adresa <http://elf.cs.pub.ro/mps/wiki/proiect/descriere>.

2.2. Context și design de bază

Flow-ul principal este următorul: utilizatorul încarcă unul sau mai multe fișiere XML obținute în urma rulării analizei de layout și rulează un analizor ierarhic pentru acestea sau să încarce un fișier XML care conține rezultatul analizei ierarhice.

2.3. Decizii de implementare

Componenta dezvoltată va fi implementată în limbajul Java.

Versiunea de Java folosită va fi Java 7.

IDE-urile folosite pentru dezvoltarea aplicației sunt Eclipse (Indigo) și NetBeans (7).

Această decizie a fost luată pentru că Java este gratuit, portabil, rapid și sigur.

2.4. Formatul fișierelor utilizate

Aplicația folosește fișiere XML ca modalitate de introducere a datelor în sistem. Datele de ieșire sunt tot fișiere XML.

Toate fișierele XML cu care lucrează componenta *Hierarchy Analysis* sunt validate de către XSD-urile puse la dispoziție la această adresă <http://elf.cs.pub.ro/mps/wiki/proiect/specificatii-xsd>.

3. Modelul datelor

3.1. Clasa HierarchyAnalysis

Scopul acestei clase este obtinerea unui fisier XML care contine rezultatul analizei ierarhice. Ii permite utilizatorului sa si aleaga modul in care va obtine fisierul de output XML.

Prima varianta: incarca unul sau mai multe fisiere XML obtinute in urma analizei de layout si ruleaza un analizor ierarhic pentru acestea. In urma rularii va obtine fisierul cu rezultatul analizei ierarhice.

Cea de a doua posibilitate este incarcarea directa a fisierului de catre utilizator si este cea recomandata.

3.1.1. Structura clasei

Membrii clasei sunt urmatoarii:

- Un grup de elemente RadioButton care permite utilizatorului selectia unui anumit mod de introducere a datelor. Exista 2 moduri posibile pentru a obtine fisierul de intrare :
 1. Selectia mai multor fisiere XML de layout si a unui analizor ierarhic. Pentru a selecta fisierele XML se va folosi JFileChooser-ul : BrowseXMLs, iar pentru selectia analizorului ierarhic se va folosi JFileChooser-ul : BrowseExec.
 2. Selectia unui singur fisier XML care contine rezultatul analizei ierarhice cu ajutorul unui JFileChooser numit : BrowseUniqueXML.
- Un JButton: ShowHierarchy. La apasarea acestui buton se va inchide fereastra curenta si se va deschide o noua fereastră care va afisa ierarhia si va permite utilizatorului sa corecteze eventualele erori.

3.1.2. Diagrama UI a clasei

The screenshot shows the 'Hierarchy Analysis Application' window. It features two radio buttons for selecting the input method. The first radio button, labeled 'Incarc fisiere de layout si un analizor ierarhic', is selected. It has two associated text fields: 'Multiple layout files...' and 'Hierarchy file...', each with a 'Select files' or 'Select file' button respectively. Below these is a 'Generate hierarchy file' button. The second radio button, labeled 'Incarca un singur fisier XML cu rezultatul analizei ierarhice', is unselected. It has one associated text field: 'XML hierarchy file...' with a 'Select file' button. At the bottom of the main area is a 'Show hierarchy' button. A 'Status Bar' is located at the very bottom of the window.

Validarea fișierelor XML de intrare se face după încărcarea acestora. Operația este realizată în background, utilizatorul fiind înștiințat dacă operația a avut succes cu ajutorul unui StatusBar, în care mesajele vor fi : Validare în curs, Fișiere XML valide sau Fișiere XML invalide.

3.2. Clasa Editor

Scopul acestei clase este afișarea diferitelor tipuri de blocuri, unirea lor , spargerea lor sau recatalogarea diferitelor blocuri.

3.2.1. Structura clasei

Membrii clasei sunt următorii :

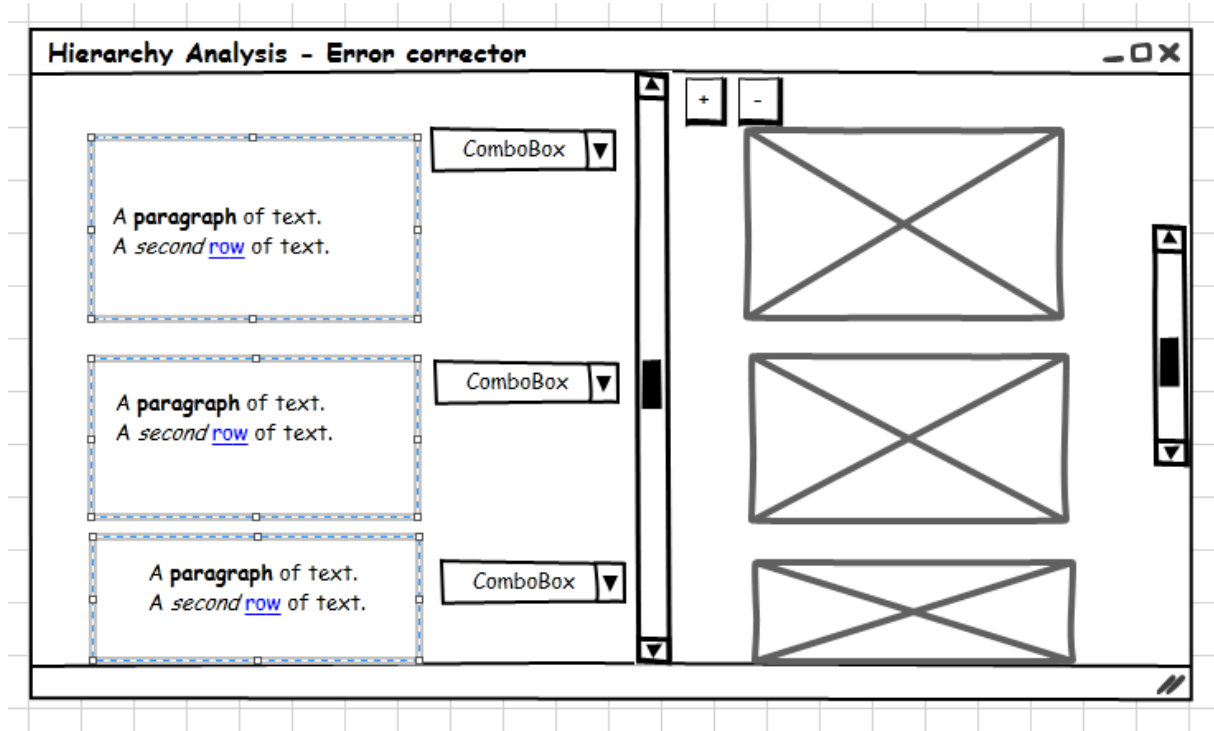
Clasa va conține două Panel-uri. Cel din stânga va conține o suprafață reprezentând pagina, pe care se vor afișa blocurile de text și tipul acestora. În dreapta fiecărui bloc se vor afla 2 ComboBox-uri, unul pentru a permite editarea respectivului bloc : split, merge, schimbare tip și celălalt pentru schimbarea fontului blocului.

Panel-ul din dreapta va conține o suprafață reprezentând pagina, pe care se vor afișa imaginile ce au fost prelucrate de către analizatorul de layout. Dacă sunt inspectate mai multe imagini , acestea vor putea fi afișate una deasupra celelalte în panel ul acesta. Imaginile sunt afișate în cadrul unui JScrollPane. De asemenea, există 2 butoane care permit zoom-ul pe o anumită pagină. Pentru cazul în care zoom-ul este prea mare, ea va putea fi inspectată cu ajutorul unui JScrollBar orizontal.

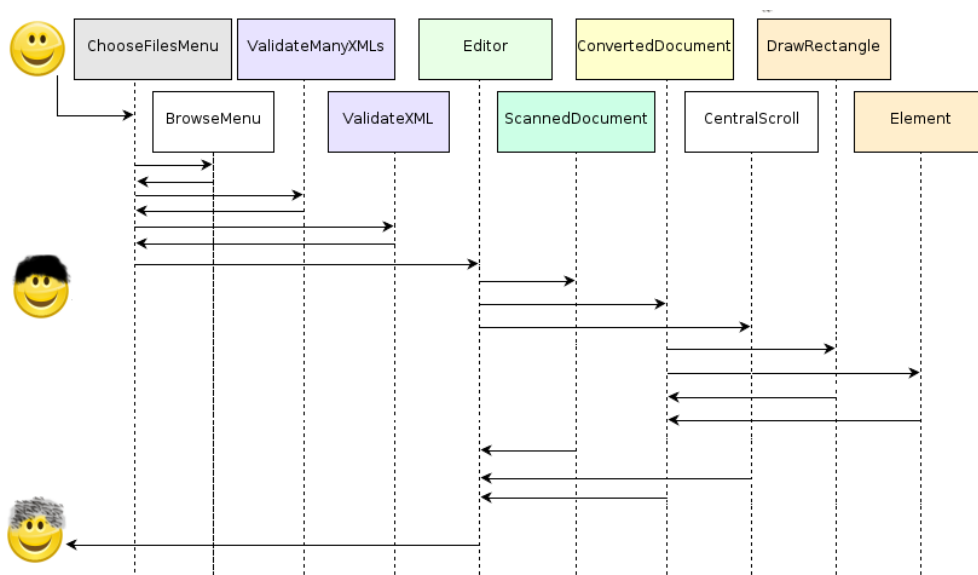
În centrul ecranului se va afla un JScrollBar care va derula în mod sincron cele două emisfere. În cazul unei desincronizări pe verticală, există în partea dreaptă un alt JScrollBar prin care se va face potrivirea.

Blocurile din partea stângă sunt instanțe ale unei clase părinte Element. Blocurile pot fi de tipul: Paragraph, Title, Subtitle, Article, Body, PageNumber, Image. Caracteristicile comune tuturor blocurilor sunt: existența unei instanțe a clasei Cadru (vizibilă în interfata grafică printr-un poligon încadrator) și conținutul propriu-zis de informație.

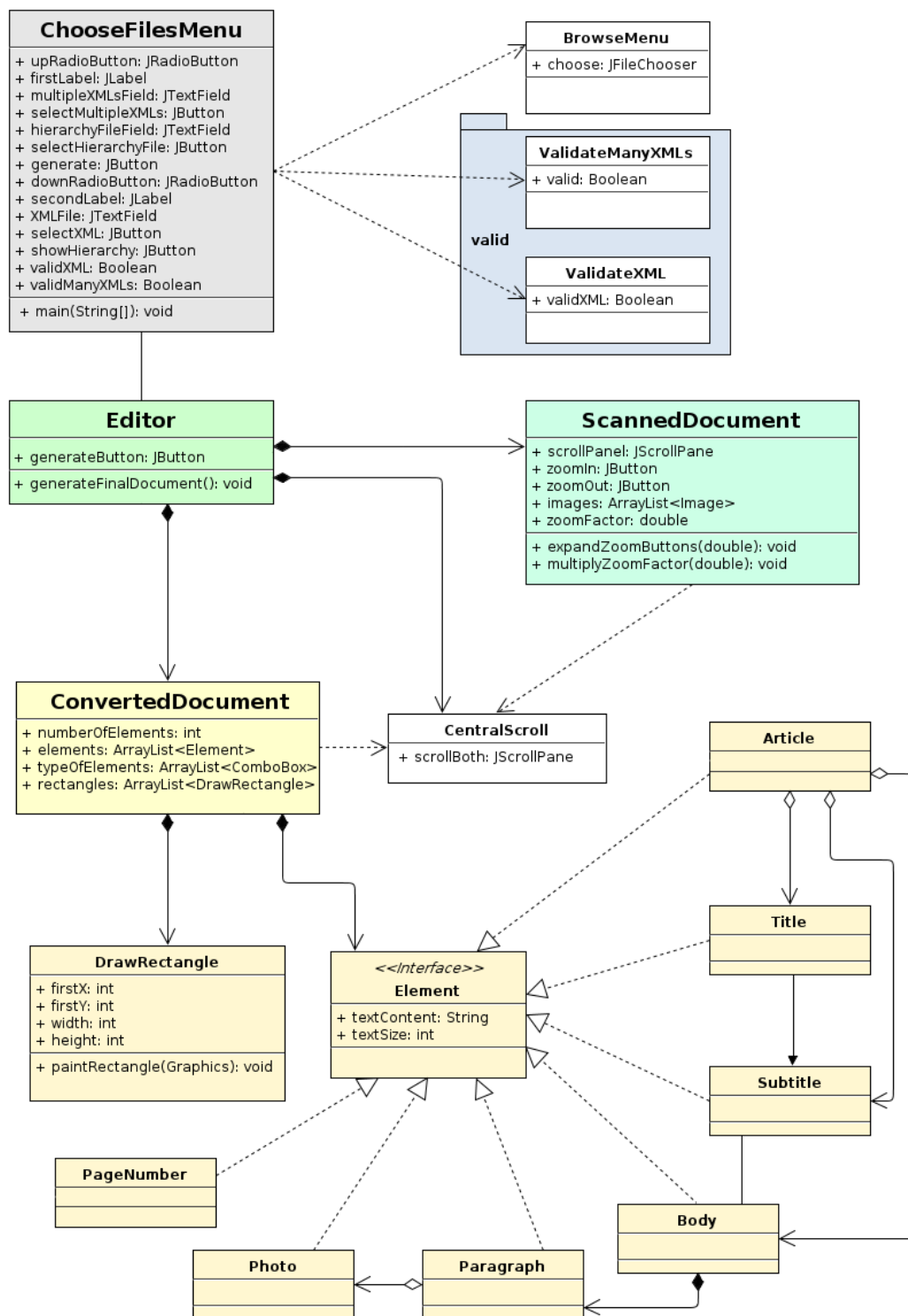
3.2.2. Diagrama UI a clasei



3.2.3. Diagrama de secventa a clasei



3.2.5. Diagrama UML a clasei



4. Structuri de date folosite

Anexa privind structurile de date folosite se gaseste la aceasta adresa
<https://docs.google.com/file/d/0B8wpiA-CzQ4-MjlvdUR4Qm5hNFU/edit>.

5. Testare

5.2. Planul de testare:

<https://docs.google.com/file/d/0B8wpiA-CzQ4-Z25zQlJvdS1FdVE/edit>

5.3. Managementul bug-urilor:

<https://docs.google.com/spreadsheet/cc?key=0AgrO3KayjJXQdDMwbmVVUTg2ZlFsb1hyVk5yZUZkWEE#gid=0>

5.4. Workflow de fixarea bug-urilor:

- Un fisier partajat in care se va raporta fiecare bug gasit de catre QE-s
- Fiecare DEV va prelua bug-urile asignate lui si le va fixa (in ordinea severitatii)
- Dupa fixare QE-ul va face regresie la bug

6. Cerințe de sistem

Procesor: Intel Pentium IV, 2 GHz

RAM: 2GB

Sistem de operare: Windows XP, Windows Vista, Windows 7, Windows 8, MAC OS X, Linux 2.4

Runtime environments: JRE 1.7

Conexiune la Internet: nu este necesara.